IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Gilbert Wolrich et al.	Art Unit:	2183
Serial No.:	09/760,509	Examiner:	Aimee J. Li
Filed:	January 12, 2001	Assignee:	Intel Corporation
Title:	METHOD AND APPARATUS FOR PROVIDING LARGE REGISTER ADDRESS SPACE WHILE MAXIMIZING CYCLETIME PERFORMANCE FOR A MULTI-THREADED REGISTER FILE SET		

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

SUPPLEMENTAL BRIEF ON APPEAL

This Appeal Brief perfects the Notice of Appeal filed on July 14, 2008 and responds to the Notification of Non-Compliant Appeal Brief mailed May 28, 2009.

(1) Real Party in Interest

Intel Corporation is the real party in interest.

(2) Related Appeals and Interferences

There are no known related appeals and/or interferences.

(3) Status of Claims

Claims 1-7, 15-26, and 30-35 are pending.

Claims 1-7, 15-26, and 30-35 are under consideration.

Claims 8-14 and 27-29 have been canceled.

Claims 1-7, 15-26, and 30-35 stand rejected.

Claim 1, 15, and 19 are in independent form.

Claims 1, 15, 19, 33, and 34 are involved directly in the appeal.

Claims 2-7, 16-18, 20-26, and 30-32, and 35 are not directly involved in the appeal but rather are involved only by virtue of their dependency from claims 1, 15, or 19.

(4) Status of Amendments

A response pursuant to 37 C.F.R. § 1.116 was filed on May 5, 2008. No claim amendments were proposed in this response. An Advisory Action mailed June 19, 2008 failed to indicate whether the May 5, 2008 response would be entered for purposes of appeal.

Since the May 5, 2008 response did not propose any claim amendments, the application is believed to be ripe for appeal regardless of whether the May 5, 2008 response is entered for purposes of appeal.

(5) Summary of Claimed Subject Matter

Execution units can include multiple microengines and allow for the execution of multiple context threads. *See, e.g., specification*, page 2, line 26-28. Such an arrangement is useful for performing tasks that are bandwidth oriented in that the microengines can have multiple threads that are simultaneously active. *See, e.g., id.*, page 2, line 24-28.

Microengines can include an execution box data path that includes an arithmetic logic unit and general purpose register set. *See, e.g., id.*, page 10, line 9-11. The general purpose register set can include a relatively large number of general purpose registers. *See, e.g., id.*, page 10, line 12-13. For example, in one implementation, a general purpose register set can include 64 general purpose registers in a first bank and 64 in a second bank. *See, e.g., id.*, page 10, line 13-14.

The addresses of the two separate banks can be interleaved on a word-by-word basis. *See, e.g., id.*, page 22, line 27-30. Each bank can be capable of performing a simultaneous read and write to two different words within its bank. *See, e.g., id.*, page 22, line 30-31. When such bank addressing is used in conjunction with register window addressing, the read bandwidth of the microengine can be satisfied using only dual ported random access memory devices. *See, e.g., id.*, page 23, line 6-8.

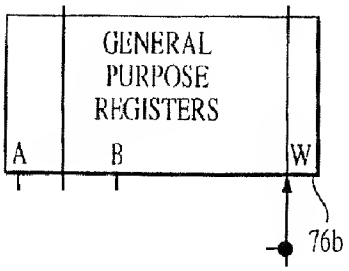
Accordingly, the following methods and devices are claimed.

Claim 1 relates to an execution unit for execution of multiple context threads (*see, e.g., id.*, page 2, line 26-28; page 10, line 6), comprising:

an arithmetic logic unit to process data for executing threads (*see, e.g., id.*, page 10, line 9-12);

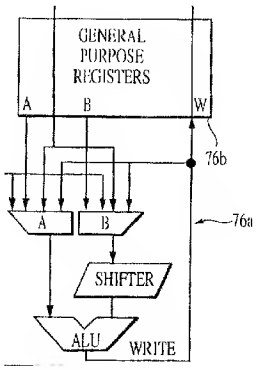
control logic to control the operation of the arithmetic logic unit (*see, e.g., id.*, page 9, line 19-20; FIG. 3, Ref. Num. 72); and

a general purpose register set to store and obtain operands for the arithmetic logic unit (*see, e.g., id.*, page 10, line 9-12), the register set comprising a plurality of two-ported random access memory devices (*see, e.g., id.*, page 23, line 6-8) assembled into banks (*see, e.g., id.*, page 10, line 13-14; page 22, line 27-30), the register set comprising two effective read ports (*see, e.g., id.*, FIG. 3, ports labeled "A," "B") and one effective write port (*see, e.g., id.*, FIG. 3, port labeled "W"),



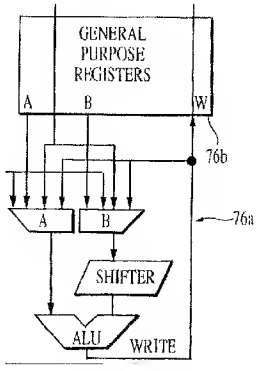
76b: general purpose register set.
See, e.g., id., p. 10, line 9-11.

wherein the effective write port comprises write ports of a pair of the two-ported random access memory devices (*see, e.g., id.*, page 22, line 27-30; page 23, line 6-8), each bank being capable of performing a read and a write to two different words in the same processor cycle (*see, e.g., id.*, page 22, line 30-31; FIG. 3, Ref. Num. 76b),



76a: arithmetic logic unit.
76b: general purpose register set.
See, e.g., id., p. 10, line 9-11.

wherein the arithmetic logic unit can write to each bank in the general purpose register set using the one effective write port. *See, e.g., id.*, FIG. 3, line labeled "WRITE" between ALU 76a and General Purpose Registers 76b; page 22, line 27-30 (describing the interleaving of addresses of different banks). *See also* Claim 1 as filed.



76a: arithmetic logic unit.
76b: general purpose register set.
See, e.g., id., p. 10, line 9-11.

Claim 15 relates to a method for executing multiple context threads, comprising:

processing data for executing threads within an arithmetic logic unit (*see, e.g., id.*, page 10, line 9-12);

operating control logic to control the arithmetic logic unit (*see, e.g., id.*, page 9, line 19-20; FIG. 3, Ref. Num. 72); and

storing and obtaining operands for the arithmetic logic unit within a general purpose register set comprising a plurality of banks of two-ported random access memory devices (*see, e.g., id.*, page 23, line 6-8), the register set comprising two effective read ports and one effective write port, wherein the effective write port comprises write ports of a pair of the two-ported random access memory devices (*see, e.g., id.*, page 22, line 27-30; page 23, line 6-8), the effective write port including a single write line to write to addresses in different banks of the plurality of banks (*see, e.g., id.*, page 22, line 27-30), and each bank being capable of performing a read and a write to two different words in the same processor cycle (*see, e.g., id.*, page 22, line 30-31; FIG. 3, Ref. Num. 76b). *See also* Claim 15 as filed.

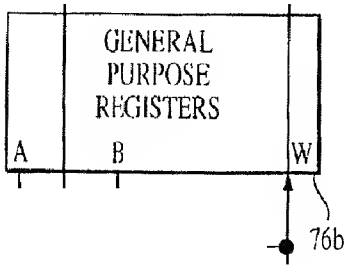
Claim 19 relates to a processor unit comprising:

an execution unit for execution of multiple context threads (*see, e.g., id.*, page 2, line 26-28; page 10, line 6), the execution unit comprising:

an arithmetic logic unit to process data for executing threads (*see, e.g., id.*, page 10, line 9-12);

control logic to control the operation of the arithmetic logic unit (*see, e.g., id.*, page 9, line 19-20; FIG. 3, Ref. Num. 72);

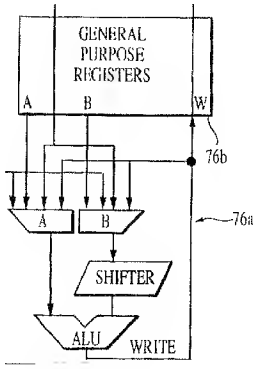
a general purpose register set to store and obtain operands for the arithmetic logic unit (*see, e.g., id.*, page 10, line 9-12), the register set comprising a plurality of two-ported random access memory devices (*see, e.g., id.*, page 23, line 6-8), the register set comprising two effective read ports (*see, e.g., id.*, FIG. 3, ports labeled "A," "B") and one effective write port (*see, e.g., id.*, FIG. 3, port labeled "W"),



76b: general purpose register set.
See, e.g., id., p. 10, line 9-11.

wherein the effective write port comprises write ports of a pair of the two-ported random access memory devices (*see, e.g., id.*, page 22, line 27-30; page 23, line 6-8); and

a data link between the arithmetic logic unit and the one effective write port of the general purpose register set, wherein the data link allows the arithmetic logic unit to write to different two-ported random access memory devices in the general purpose register set through the one effective write port. *See, e.g., id.*, page 22, line 27-30; FIG. 3, line labeled "WRITE" between ALU 76a and General Purpose Registers 76b. *See also* Claim 19 as filed.



76a: arithmetic logic unit.
76b: general purpose register set.
See, e.g., id., p. 10, line 9-11.

Claim 33 relates to the execution unit of claim 1 wherein memory addresses of the pair of the two-ported random access memory devices are interleaved. *See, e.g., id.*, page 22, line 27-30.

Claim 34 relates to the processor of claim 19 wherein memory addresses of the pair of the two-ported random access memory devices are interleaved. *See, e.g., id.*, page 22, line 27-30.

(6) Grounds of Rejection to be Reviewed on Appeal

As set forth in the following concise statements, the following grounds for rejection are presented for review on appeal:

Ground 1: Whether claims 1-5 and 15-22 are properly rejected under 35 U.S.C. 103(a) as obvious over U.S. Patent No. 5,933,627 to Parady (hereinafter "Parady") and U.S. Patent Number 5,787,454 to Rohlman (hereinafter "Rohlman").

Ground 2: Whether claims 6-7 and 23-25 are properly rejected under 35 U.S.C. 103(a) as obvious over Parady, Rohlman, and the document entitled "Register Relocation: Flexible Contexts for Multithreading," by Waldspurger et al. (hereinafter "Waldspurger").

Ground 3: Whether claim 26 is properly rejected under 35 U.S.C. 103(a) as obvious over Parady, Rohlman, Waldspurger, and U.S. Patent No. 5,509,130 to Trauben et al. (hereinafter "Trauben").

(7) Argument

Ground 1: Rejections under 35 U.S.C. § 103(a) as obvious over Parady and Rohlman

Claim 1 relates to an execution unit for execution of multiple context threads. The execution unit includes a general purpose register set to store and obtain operands for the

arithmetic logic unit. The register set includes a plurality of two-ported random access memory devices assembled into banks. The register set includes two effective read ports and one effective write port. The effective write port includes write ports of a pair of the two-ported random access memory devices. Each bank is capable of performing a read and a write to two different words in the same processor cycle. An arithmetic logic unit can write to each bank in the general purpose register set using the one effective write port.

The rejection of claim 1 contends that it would have been obvious for one of ordinary skill to have combined Parady and Rohlman to have arrived at the recited subject matter. Applicant respectfully disagrees and submits that claim 1 is not obvious over Parady and Rohlman.

In this regard, FIGS. 1 and 3 of Parady relate to an Ultrasparc microprocessor which can be modified to incorporate Parady's invention. *See, e.g., Parady*, col. 2, line 45-48; 51-53; 66-67. This Ultrasparc microprocessor includes integer registers 48 (also referred to as "integer register file 48.") *See, e.g., id.*, FIG. 1. Parady does not regard integer register files 48 as being inventive. Rather, Parady regards program address registers 110, and the functionality provided thereby, as inventive. *See, e.g., id.*, para. 3, line 50-55.

Perhaps because of this, Parady provides limited details regarding the structure of integer registers 48. The following two excerpts set forth the description of integer registers 48 in Parady.

> "The first three functional units, the load/store unit 32 and the two integer ALU units 34 and 36, share a set of integer registers 48." *See Parady*, col. 3, line 18-20.

> "Integer register file 48 is divided up into four register files to support threads 0-3. Similarly, floating point register file 50 is broken into four register files to support threads 0-3. This can be accomplished either by providing physically separate groups of registers for each thread, or alternately by providing separate register windows for each thread." *See Parady*, col. 3, line 44-49.

Also, FIG. 1 of Parady depicts integer registers 48 and labels them with the following text "integer registers (8 windows, 7 read, 3 write ports)."

Against this backdrop, the rejection contends that Parady's integer registers 48 comprise a plurality of two-ported random access memory devices assembled into banks that are each capable of performing a read and a write to two different words with two ports in the same processor cycle where an arithmetic logic unit can write to each bank in the general purpose register set using the one effective write port. The rejection also contends that, based on Rohlman's arrangement of memory cells, these banks are not only amenable but also obvious to modify to have two

effective read ports and one effective write port, where the effective write port includes write ports of a pair of the two-ported random access memory devices.

Applicant respectfully submits that there is no support for these contentions. For example, there is no reason to believe that Parady's integer registers 48 are anything other than standard multiported registers. Indeed, Parady at least suggests that they are standard multiported registers in describing that integer registers 48 must be "divided up." Integer registers 48 are thus understood to start as a whole unit—such as standard multiported register. Such a whole unit is subsequently divided up rather than "assembled into banks." Even if this suggestion is not definitive, it is pure speculation to contend that Parady's integer registers 48 are not standard multiported registers but a plurality of two-ported random access memory devices assembled into banks in the manner recited in claim 1.

Indeed, the only mention of two-ported register files made in Parady is found at col. 5, line 30-43, in which Parady discusses dual-ported shadow register files. *See Parady*, col. 5, line 30-43; FIG. 7. Parady describes that addressing these dual-ported shadow register files requires a switch 192 or an enable path 194. *See, e.g., Parady*, col. 5, line 38-44. Parady thus was plainly aware of the existence of two-ported register

files and their potential use in data storage. However, Parady fails to describe or suggest that even these dual-ported shadow register files be assembled into banks, or that the register set includes two effective read ports and one effective write port, or that the effective write port includes write ports of a pair of the two-ported random access memory devices, as recited in claim 1. Instead, Parady addresses different two-ported shadow register files using switch 192 or enable path 194.

Returning to integer registers 48, as best understood, the contention that Parady's integer registers 48 are somehow "assembled into banks" is based on Parady's description that integer registers 48 can be divided up into four register files "by providing physically separate groups of registers for each thread, or alternately by providing separate register windows for each thread." *See Parady*, col. 3, line 44-49.

Applicant respectfully disagrees for several reasons. For example, any provision of physically separate groups of registers would presumably be done with the same standard multiported registers that form integer registers 48. Moreover, there is no reason to believe that Parady would depart from the approach to providing physically separate registers that Parady uses with the physically separate two-ported shadow register files, namely, using a switch or enable path.

As for providing separate register windows, windowing does not assemble two ported memory devices into banks. Rather, windowing shifts an active register window by a certain number of registers. *See, e.g.,* http://en.wikipedia.org/wiki/Register_window. Indeed, the Office actions are understood to acknowledge the distinction between banks of memory devices and windowing. *See, e.g., Office action mailed March 13, 2008*, para. 9 (rejecting claims 2 and 16).

Accordingly, not only do Parady's integer registers 48 not comprise a plurality of two-ported random access memory devices, the constituent devices Parady's integer registers 48 are also not assembled into banks, as recited in claim 1.

Rohlman does nothing to remedy these deficiencies in Parady. Instead, Rohlman describes memory cells that are each capable of storing a single bit, reading a single bit, and writing a single bit. *See Rohlman*, col. 6, line 44-65. Groups of these memory cells can be assembled into banks in a Re-Order Buffer (ROB) and addressed using interleaving. *See Rohlman*, col. 7, line 20-29. However, each of these banks is written using a single separate data input 412, 414, 416, 418. *See Rohlman*, FIG. 7; col. 7, line 20-29. Each of these banks is read using a single separate data output 442, 4144, 446, 448. *See Rohlman*, FIG. 7; col. 7, line 56-62.

Since none of data inputs 412, 414, 416, 418 can be used to write to multiple banks in Rohlman's Re-Order Buffer (ROB), these memory cell banks are not a plurality of two-ported random access memory devices assembled into banks that are each capable of performing a read and a write to two different words with two ports in the same processor cycle where an arithmetic logic unit can write to each bank in the general purpose register set using the one effective write port, as recited.

Indeed, the rejection does not contend otherwise. Instead, the rejection ignores Rohlman's description of how groups of memory cells are to be assembled into banks and instead contends:

> "the language used calls each group of memory cells a memory bank, however, for purposes of the rejection Rohlman's memory cells are equated to the claimed memory bank, since Rohlman's memory cells meet the claimed meaning of memory bank." *See Office Action mailed March 13, 2008*, para. 8 (emphasis added).

Thus, as best understood, the rejection does not consider a group of memory cells to be the recited memory bank. Instead, the rejection considers *individual memory cells* to be a bank that is assembled from a plurality of two-ported random access memory devices. Further, the rejection contends that those of ordinary skill would not only ignore Rohlman's description that groups of memory cells are banks, they would also ignore

Rohlman's description as to how these groups of memory cells are to be assembled into banks and instead be motivated to modify banks of two-ported memory devices to have two effective read ports and one effective write port based on the way that Rohlman assembles individual memory cells.

Applicant respectfully submits that this is improper hindsight-based reconstruction of the claimed technology. There is no reason to believe that those of ordinary skill would ignore the way that Rohlman writes to groups of memory cells and instead find inspiration in the way that Rohlman writes to individual cells when writing to banks that have been assembled from two-ported memory devices. Indeed, Rohlman himself was plainly aware of his own approach to writing to individual cells and yet did not chose to apply it when writing to groups of cells. There is no reason to believe that those of ordinary skill would find the suggested modification obvious when Rohlman did not arrive at the suggested modification.

Thus, Paraday and Rohlman both fail to describe or suggest the recited subject matter and would not lead one of ordinary skill to the recited subject matter even if combined. Accordingly, claim 1 is not obvious over Paraday and Rohlman. Applicant respectfully requests that the rejections of claim 1, and the claims dependent therefrom, be withdrawn.

Claim 33 relates to the execution unit of claim 1 wherein memory addresses of the pair of the two-ported random access memory devices are interleaved.

Paraday and Rohlman neither describe nor suggest that memory addresses of the pair of the two-ported random access memory devices be interleaved. In this regard, Paraday does not include any detail as to the addressing of what are understood to be the standard multiported registers that form integer registers 48.

As for Rohlman, as discussed above, the rejection of claim 1 is based on the contention that Rohlman's individual memory cells are banks assembled from a plurality of two-ported random access memory devices. However, the rejection of claim 33 ignores the basis on which the parent claim is rejected and instead contends that those of ordinary skill would find it obvious to have interleaved banks assembled from a plurality of two-ported random access memory devices based on the way groups of individual memory cells are interleaved in Rohlman.

Applicant respectfully submits that it would not have been obvious for those of ordinary skill to have relied upon such logical inconsistencies in arriving at the subject matter recited in claim 33. Once again, this is hindsight-based

reasoning in which features of different arrangement are drawn out of context and assembled using applicant's disclosure as a guide.

Accordingly, claim 33 is not obvious over Paraday and Rohlman on this basis as well. Applicant respectfully requests that the rejections of claim 33 be withdrawn.

Claim 15 relates to a method that includes storing and obtaining operands for an arithmetic logic unit within a general purpose register set comprising a plurality of banks of two-ported random access memory devices. The register set includes two effective read ports and one effective write port. The effective write port comprises write ports of a pair of the two-ported random access memory devices. The effective write port including a single write line to write to addresses in different banks of the plurality of banks. Each bank is capable of performing a read and a write to two different words in the same processor cycle.

The rejection of claim 15 briefly states that "claim 15 is rejected for the same reasons as claim 1."

Applicant respectfully traverses the rejection to the extent that the rejection of claim 15 suffers from the deficiencies discussed above.

Further, claim 15 recites that an effective write port includes a single write line to write to addresses in different banks of the plurality of banks. Applicant respectfully submits that such a single write line further illustrates the unreasonableness of considering Parady's "physically separate groups of registers" to be banks assembled from a plurality of two-ported random access memory devices. In this regard, the "physically separate groups of registers" would appear to require physically separate write lines. Parady neither describes nor suggest that a single write line can be used to write to addresses in the physically separate groups of registers.

Accordingly, claim 15 is not obvious over Paraday and Rohlman. Applicant respectfully requests that the rejections of claim 15, and the claims dependent therefrom, be withdrawn.

Claim 19 relates to a processor unit that includes a general purpose register set and a data link. The general purpose register set that includes a plurality of two-ported random access memory devices. The register set includes two effective read ports and one effective write port. The effective write port includes write ports of a pair of the two-ported random access memory devices.

The data link is between the arithmetic logic unit and the one effective write port of the general purpose register set. The data link allows the arithmetic logic unit to write to different two-ported random access memory devices in the general purpose register set through the one effective write port.

The rejection of claim 19 suffers from many of the same deficiencies discussed above with regard to claim 1. For example, the rejection contends that Parady's integer registers 48 comprise a plurality of two-ported random access memory devices. As another example, the rejection contends that, based on Rohlman's arrangement of memory cells, this register set is not only amenable but also obvious to modify to have two effective read ports and one effective write port, where the effective write port includes write ports of a pair of the two-ported random access memory devices.

Applicant respectfully disagrees. As discussed above, it is pure speculation to contend that Parady's integer registers 48 are not standard multiported registers but a plurality of two-ported random access memory devices. Indeed, Parady at least suggests the opposite in describing that integer registers 48 can be "divided up," e.g., "by providing physically separate groups of registers for each thread."

Further, there is no reason to believe that those of ordinary skill would find inspiration in Rohlman's individual memory cells to modify Parady's integer registers 48 to have two effective read ports and one effective write port, where the effective write port includes write ports of a pair of the two-ported random access memory devices. For example, Rohlman was plainly aware of his individual memory cells but yet chose to use separate write lines to when writing to groups of cells. Absent the use of hindsight, there is no reason to believe that those of ordinary skill would adopt an arrangement that Rohlman did not.

Accordingly, claim 15 is not obvious over Paraday and Rohlman. Applicant respectfully requests that the rejections of claim 15, and the claims dependent therefrom, be withdrawn.

Claim 34 relates to the processor of claim 19 wherein memory addresses of the pair of the two-ported random access memory devices are interleaved.

Paraday and Rohlman neither describe nor suggest that memory addresses of the pair of the two-ported random access memory devices be interleaved. In this regard, Paraday does not include any detail as to the addressing of what are understood to be the standard multiported registers that form integer registers 48.

As for Rohlman, as discussed above, the rejection of claim 19 is based on the contention that Rohlman's individual memory cells are banks assembled from a plurality of two-ported random access memory devices. However, the rejection of claim 34 ignores the basis on which the parent claim is rejected and instead contends that those of ordinary skill would find it obvious to have interleaved banks assembled from a plurality of two-ported random access memory devices based on the way groups of individual memory cells are interleaved in Rohlman.

Applicant respectfully submits that it would not have been obvious for those of ordinary skill to have relied upon such logical inconsistencies in arriving at the subject matter recited in claim 34. Once again, this is hindsight-based reasoning in which features of different arrangements are drawn out of context and assembled using applicant's disclosure as a guide.

Accordingly, claim 34 is not obvious over Paraday and Rohlman on this basis as well. Applicant respectfully requests that the rejections of claim 34 be withdrawn.

Please apply any charges or credits to Deposit Account No. 06-1050.

Respectfully submitted,

Date: June 29, 2009

/John F. Conroy, Reg. #45,485/
John F. Conroy
Reg. No. 45,485

Fish & Richardson P.C.
Regus Business Center
Landsberger Strasse 155
Munich 80687
Germany
Telephone: 011 49 (89) 57959 - 125
Facsimile: (877) 769-7945

JFC/jhg
14007509.doc

Appendix of Claims

1. An execution unit for execution of multiple context threads, comprising:

an arithmetic logic unit to process data for executing threads;

control logic to control the operation of the arithmetic logic unit; and

a general purpose register set to store and obtain operands for the arithmetic logic unit, the register set comprising a plurality of two-ported random access memory devices assembled into banks, the register set comprising two effective read ports and one effective write port, wherein the effective write port comprises write ports of a pair of the two-ported random access memory devices, each bank being capable of performing a read and a write to two different words in the same processor cycle,

wherein the arithmetic logic unit can write to each bank in the general purpose register set using the one effective write port.

2. The execution unit of claim 1 wherein the register set is logically partitioned into a plurality of relatively addressable windows.

3. The execution unit of claim 2 wherein the number of windows of the register set is related to the number of threads that execute in the processor.

4. The execution unit of claim 1 wherein relative addressing allows an executing thread to access the register set relative to the starting point of a window.

5. The execution unit of claim 1 wherein the register set is absolutely addressable, where the register set may be accessed for an executing thread by providing an exact address.

6. The execution unit of claim 1 wherein the control logic further comprises:

context event switching logic fed by signals from a plurality of shared resources, the signals causing the context event switching logic to indicate that threads are either available or unavailable for execution.

7. The execution unit of claim 6 wherein the control logic addresses a first set of memory locations for storing a list of available threads that correspond to threads that are ready to be executed and a second set of memory locations for storing a list of unavailable threads that are not ready to be executed.

Claims 8-14. (Canceled)

15. A method for executing multiple context threads, comprising:

processing data for executing threads within an arithmetic logic unit;

operating control logic to control the arithmetic logic unit; and

storing and obtaining operands for the arithmetic logic unit within a general purpose register set comprising a plurality of banks of two-ported random access memory devices, the register set comprising two effective read ports and one effective write port, wherein the effective write port comprises write ports of a pair of the two-ported random access memory devices, the effective write port including a single write line to write to addresses in different banks of the plurality of banks, and each bank being capable of performing a read and a write to two different words in the same processor cycle.

16. The method of claim 15 wherein the register set is relatively addressable.

17. The method of claim 15 further comprising:

arranging the register set into a number of windows according to the number of threads that execute.

18. The method of claim 17 wherein storing and obtaining further comprises:

addressing the register set for an executing thread by providing a relative register address that is relative to the starting point of a window.

19. A processor unit comprising:

an execution unit for execution of multiple context threads, the execution unit comprising:

an arithmetic logic unit to process data for executing threads;

control logic to control the operation of the arithmetic logic unit;

a general purpose register set to store and obtain operands for the arithmetic logic unit, the register set comprising a plurality of two-ported random access memory devices, the register set comprising two effective read ports and one effective write port, wherein the effective write port comprises write ports of a pair of the two-ported random access memory devices; and

a data link between the arithmetic logic unit and the one effective write port of the general purpose register set, wherein the data link allows the arithmetic logic unit to write to different two-ported random access memory devices in the general purpose register set through the one effective write port.

20. The processor of claim 19 wherein the register set is logically partitioned into a plurality of relatively addressable windows, where the number of windows of the register set is related to the number of threads that execute in the processor.

21. The processor of claim 20 wherein relative addressing allows an executing thread to access the register set relative to the starting point of a window.

22. The processor of claim 20 wherein the register set is absolutely addressable, where the register set may be accessed for an executing thread by providing an exact address.

23. The processor of claim 20 further comprising:

a set of memory locations for storing a list of available threads that are ready to be executed;

a set of memory locations for storing a list of unavailable threads that are not ready to be executed; and

context event switching logic fed by signals from a plurality of shared resources, the signals causing the context event switching logic to indicate which threads are either available or unavailable for execution.

24. The processor of claim 23 wherein execution of a context swap instruction causes a currently running thread to be swapped out to the list of unavailable threads and a thread from the list of available threads to begin execution within a single execution cycle.

25. The processor of claim 23 wherein execution of a context swap instruction specifies one of the signals and upon receipt of the specified signal causes a swapped out thread to be stored in the available thread memory set.

26. The processor of claim 23 wherein execution of a context swap instruction specifies a defer_one operation which causes execution of one more instruction and then causes a current context to be swapped out.

Claims 27-29. (Canceled)

30. The execution unit of claim 1 wherein the register set comprises a first number *n* of two-ported random access memory devices, a second number *r* of effective read ports, and a third number *w* of effective write ports, where n $\geq$ 2, 2 $\leq$ *r* $\leq$ *n*, and 2 $\leq$ *w* $\leq$ *n-1*.

31. The method of claim 15 wherein storing and obtaining operands comprises storing and obtaining operands within the general purpose register comprising a first number *n* of two-ported random access memory devices, a second number *r* of effective read ports, and a third number *w* of effective write ports, where n $\geq$ 2, 2 $\leq$ *r* $\leq$ *n*, and 2 $\leq$ *w* $\leq$ *n-1*.

32. The execution unit of claim 19 wherein the general purpose register set comprises a first number *n* of two-ported random access memory devices, a second number *r* of effective read ports, and a third number *w* of effective write ports, where n $\geq$ 2, 2 $\leq$ *r* $\leq$ *n*, and 2 $\leq$ *w* $\leq$ *n-1*.

33. The execution unit of claim 1 wherein memory addresses of the pair of the two-ported random access memory devices are interleaved.

34. The processor of claim 19 wherein memory addresses of the pair of the two-ported random access memory devices are interleaved.

35. The method of claim 15 wherein storing the operands for the arithmetic logic unit comprises writing to interleaved memory addresses of the pair of the two-ported random access memory devices over the single write line.

Evidence Appendix

None.

Related Proceedings Appendix

None.